Retargeted Matrix Factorization for Collaborative Filtering

Oluwasanmi Koyejo
Electrical and Computer Engineering Dept.,
University Of Texas, Austin.
sanmi.k@utexas.edu

Sreangsu Acharyya
Electrical and Computer Engineering Dept.,
University Of Texas, Austin.
sreangsu@utexas.edu

Joydeep Ghosh
Electrical and Computer Engineering Dept.,
University Of Texas, Austin.
ghosh@ece.utexas.edu

ABSTRACT

This paper introduces retargeted matrix factorization (R-MF); a novel approach for learning the user-wise ranking of items in the context of collaborative filtering. R-MF learns to rank by "retargeting" the item ratings of each user, searching for a monotonic transformation of the ratings that results in a better fit while preserving the ranked order of each user's ratings. The retargeting is combined with an underlying matrix factorization regression model that couples the user-wise rankings to exploit shared low dimensional structure. We show that R-MF recovers a unique solution under mild conditions, and propose a simple and efficient optimization scheme that alternates between retargeting the ratings subject to ordering constraints, and matrix factorization regression. The retargeting step is independent for each user, and is trivially parallelized. The ranking performance of retargeted matrix factorization is evaluated on benchmark movie recommendation datasets and results in superior ranking performance compared to collaborative filtering algorithms specifically designed to optimize ranking metrics.

Categories and Subject Descriptors

H3.3 [**Information Search and Retrieval**]: Information filtering—*Collaborative Filtering*; G3 [**Probability and Statistics**]: Correlation and regression analysis

General Terms

Algorithms, Experimentation

Keywords

Matrix factorization; Collaborative filtering; Learning to rank

1. INTRODUCTION

The focus of research in the recommender systems literature has begun to shift from algorithms and metrics for optimizing *regression* accuracy to learning and measuring the *ranking* of items for each user. Learned rating scores are not shown in most practical deployments of recommender systems. Instead, the user is shown a few of the top items as ordered by the recommender system. This means that although regression metrics such as root mean square error (RMSE) and mean absolute error (MAE) are easier to optimize, ranking metrics such as normalized discounted cumulative gain (NDCG) and expected reciprocal return (ERR) [18] are a more accurate reflection of how recommender systems are used in practice [9].

This paper proposes retargeted matrix factorization (R-MF), a novel approach for learning the user-wise ranking of items inspired by the study of learning to rank (LETOR) in the the information retrieval literature [18]. There, initial approaches using point-wise ranking models [8] were replaced by *pair-wise* models [10], and are now being superseded by *list-wise* ranking models [6, 2]. The list-wise approach learns a ranking model for the entire set of items and has gained prominence in the LETOR literature with strong theoretical guarantees and superior empirical performance [19]. Retargeted matrix factorization (R-MF) transforms the matrix factorization (MF) model into a user-wise ranking model by adjusting the rating values based on the regression fit. These adjustments are constrained to monotonic transformations that preserve the order of the user ratings. We show that R-MF inherits useful statistical and optimization theoretic properties when the loss function for the matrix factorization is a *Bregman divergence* [4], a family of divergences that includes such popular loss functions such as squared loss, Kullback-Leibler (KL) divergence and the I-divergence (also known as the generalized KL divergence). In the context of learning to rank, Bregman divergences are further motivated as the *unique* family of cost functions that are strongly consistent with the NDCG metric [19].

The main contributions of this paper are as follows:

- We propose retargeted matrix factorization (R-MF), a user-wise ranking model for collaborative filtering based on list-wise learning to rank. R-MF jointly searches over monotonic transformations of ratings for each user and an underlying matrix factorization regression that exploits shared structure in the user lists.

- We show that when squared loss is used as the regression loss function, the solution of the retargeted matrix factorization is unique in terms of retargeted scores and the regression matrix under verifiable conditions.

- We propose a simple optimization scheme that alternates between the retargeting step and the regression step. The retargeting step is independent for each user and can be solved in parallel. The user and item factors can be estimated using any number of matrix factorization algorithms.

RecSys'13, October 12–16, 2013, Hong Kong, China.
Copyright 2013 ACM 978-1-4503-2409-0/13/10 ...$15.00.
http://dx.doi.org/10.1145/2507157.2507185.

- Retargeted matrix factorization is evaluated on benchmark movie recommendation datasets and compared to collaborative filtering algorithms specifically designed to optimize ranking metrics.

This paper is organized as follows, we begin with an overview of related work in Section 1.1. This is followed by a discussion of the modeling approach and some of the resulting properties in Section 2. We propose a simple alternating optimization scheme in Section 3. Experimental results are discussed in Section 4 and we conclude in Section 5.

Notation: Vectors are denoted by bold lower case letters, matrices are capitalized. $\boldsymbol{x}^\top$ denotes the transpose of the vector $\boldsymbol{x}$, $||\boldsymbol{x}||$ denotes the L_2 norm. A vector $\boldsymbol{x}$ is defined to be in *descending order* if $x_i \geq x_j$ when $i > j$, the set of such vectors is denoted by $\mathcal{R}\!\downarrow$. Vector $\boldsymbol{x}$ is isotonic with $\boldsymbol{y}$ if $x_i \geq x_j$ implies $y_i \geq y_j$. The unit simplex is denoted by Δ and Δ_ϵ denotes the subset of an unit simplex such that each of its members are component-wise bounded away from 0 by ϵ. The positive orthant is denoted by $\mathcal{R}^d_+$ and $\mathcal{R}^d_\epsilon$ denotes its subset such that each of its members are component-wise bounded away from 0 by ϵ.

1.1 Related Work

Models for user-wise ranking have been studied by several researchers in the recommender systems literature. Point-wise models applied to collaborative filtering predict user ratings using regression or classification methods, the final ranking is defined by the ordering of the regression scores. It is often difficult to extract a clear relationship between the regression scores and the item rankings. Cremonesi et al. [9] showed that methods trained to optimize regression metrics may not be effective for top-k recommendation. Steck et al. [24] proposed modifications of matrix factorization methods motivated by top-k ranking performance. A related class of point-wise ordinal regression models have also been proposed. These models are optimized so that user ratings are correctly placed in ordered bins corresponding to the different rating levels. For instance, maximum margin matrix factorization (MMMF) [23] jointly optimized the matrix factorization and the user rating bins using the Hinge loss, CoFi$^{\text{RANK}}$ -ordinal [26] used bundle methods with squared loss for fast optimization of the user and item factors and ordinal regression bins, and Ordrec [14] combined ordinal logistic regression with matrix factorization.

Ranking performance may be improved further by using a pairwise approach. In this case, the model is trained to order each pair of items [17]. Balakrishnan et al. [3] proposed a pairwise classification approach for collaborative ranking. The authors showed effective ranking performance as measured using the NDCG metric. Pair-wise ranking results in a model with computational cost that is quadratic in the number of items to be ranked. This is a significant increase in computational cost and may be prohibitive in large scale recommender systems with millions of ratings[1]. The learned pairwise orders must also be converted into a fully ordered list (at additional computational cost). Of significant concern is the fact that pair-wise orders are not necessarily transitive. Hence, orders over pairs of items may not directly translate into an ordered list. For example, given three items $\{a, b, c\}$, there is no consistent ordered list if the pair-wise relations are given by $\{a > b\}, \{b > c\}, \{c > a\}$.

List-wise approaches avoid the computational and consistency hazards of pair-wise methods. In addition, list-wise methods often have stronger statistical and optimization theoretic guarantees [19, 2], and superior empirical performance. List-wise models applied to collaborative filtering are known as user-wise ranking models. CoFi$^{\text{RANK}}$ [25], a popular approach for user-wise collaborative filtering, trains a matrix factorization model to optimize a bound of the NDCG metric. Shi et al. [21] adapted the list-wise ranking algorithm proposed in [6] to collaborative filtering by replacing the underlying linear regression model with a matrix factorization regression model.

[1]Sub-sampling has been proposed as an approach to reduce this computational burden [3]. However, we note that any computational savings also apply to competing approaches.

Table 1: Examples of identically separable (IS) Bregman divergences. Squared loss (top) and KL divergence (bottom).

$\boldsymbol{\phi}(\boldsymbol{x})$	$D_\phi\big(\boldsymbol{x}\big\|\big\|\boldsymbol{y}\big)$
$\frac{1}{2}\|\|\boldsymbol{x}\|\|^2_W$	$\frac{1}{2}\|\|\boldsymbol{x}-\boldsymbol{y}\|\|^2_2$
$\sum_i x_i \log x_i,\ \boldsymbol{x} \in \Delta$	$\mathrm{KL}\,(x\|\|y) = \sum_i x_i \log(\frac{x_i}{y_i})$

The outlined related work is focused on models for optimizing top-k ranking performance and not on regression metrics such as RMSE / MAE [13]. Further, there is a large literature on the use of side information such as features [5] and graphs [15] for recommender systems. The use of such side information enables the model to be used for *cold start* i.e. predicting the rankings of users with no training ratings. Such extensions are not discussed here in detail and are left as a subject for future work.

The proposed framework includes a choice of loss function in the family of Bregman divergences. A unified approach for matrix factorization with Bregman divergence loss functions was proposed by Singh and Gordon [22], showing that several well known methods such as non-negative matrix factorization (NMF), weighted singular value decomposition (SVD), maximum margin matrix factorization (MMMF), probabilistic latent semantic indexing (pLSI) and other related models can be posed as special cases of this framework.

1.2 Preliminaries

Let $\boldsymbol{\phi} : \Theta \mapsto \mathbb{R},\ \Theta = \operatorname{dom}\boldsymbol{\phi} \subseteq \mathbb{R}^d$ be a strictly convex, closed function, differentiable on $\operatorname{int}\Theta$. The corresponding *Bregman divergence* $D_\phi\big(\cdot\big\|\big\|\cdot\big) : \operatorname{dom}(\boldsymbol{\phi}) \times \operatorname{int}(\operatorname{dom}(\boldsymbol{\phi})) \mapsto \mathbb{R}_+$ is defined as $D_\phi\big(\boldsymbol{x}\big\|\big\|\boldsymbol{y}\big) \triangleq \boldsymbol{\phi}(\boldsymbol{x}) - \boldsymbol{\phi}(\boldsymbol{y}) - \langle \boldsymbol{x}-\boldsymbol{y}, \nabla\boldsymbol{\phi}(\boldsymbol{y})\rangle$. From strict convexity it follows that $D_\phi\big(\boldsymbol{x}\big\|\big\|\boldsymbol{y}\big) \geq 0$ and $D_\phi\big(\boldsymbol{x}\big\|\big\|\boldsymbol{y}\big) = 0$ iff. $\boldsymbol{x} = \boldsymbol{y}$. Bregman divergences are (strictly) convex in their first argument, but not necessarily convex in their second.

In this paper we only consider functions of the form $\boldsymbol{\phi}(\cdot) : \mathbb{R}^n \ni \boldsymbol{x} \mapsto \sum_i \phi(x_i)$ that are sums of *identical* scalar convex functions applied to each component. We refer to this class as *identically separable* (**IS**). This class has properties particularly suited to ranking. Squared loss and Kullback-Leibler divergence (KL) are members of this family (Table 1).

The *Legendre conjugate* $\psi(\cdot)$ of the function $\phi(\cdot)$ is defined as $(\phi)^*(\boldsymbol{x}) \triangleq \psi(\boldsymbol{x}) \triangleq \sup_{\boldsymbol{\lambda}}(\langle\boldsymbol{\lambda}, \boldsymbol{x}\rangle - \phi(\boldsymbol{\lambda}))$. If $\phi(\cdot)$ is a convex function of Legendre type [20] (as will always be the case in this paper), $(()^*)^*(\cdot) = \phi(\cdot)$ and $(\nabla\phi(\cdot))^{-1} = \nabla\psi(\cdot)$ is a one to one mapping.

2. LEARNING TO RECOMMEND

Let w_{ij} be a binary variable that denotes whether user i has rated item j ($w_{ij} = 1$) or not ($w_{ij} = 0$). We denote the true ratings by Y_{ij} and predicted ratings by $\hat{Y}_{ij}$. Let $\mathcal{U}$ and $\mathcal{V}$ denote the set of users and items respectively. Let $\mathcal{V}_i$ denote the set of items rated by user i and $|\mathcal{V}_i|$ its cardinality. The item recommendation task

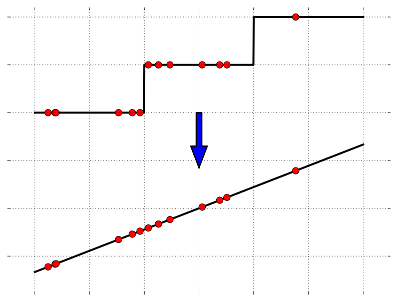


Figure 1: Monotone retargeting [2] searches for an order preserving transformation of the target scores that may be easier for the regressor to fit.

consists of learning the user's taste from the training ratings. A popular technique for item recommendation is matrix factorization (MF) [13]. In MF, the predictions take the form:

$$\hat{Y}_{ij} = \langle \boldsymbol{u}_i, \boldsymbol{v}_j \rangle \,, \tag{1}$$

where $\boldsymbol{u}_i \in \mathbb{R}^d$ is called a user factor for user i and $\boldsymbol{v}_j \in \mathbb{R}^d$ is the item factor for item j. Further, let $U \in \mathbb{R}^{|\mathcal{U}| \times d}$ with $U(i) = \boldsymbol{u}_i$ represent the collected user factors and let $V \in \mathbb{R}^{|\mathcal{V}| \times d}$ with $V(j) = \boldsymbol{v}_j$ represent the collected item factors. Note that this factorization constrains the rank of the matrix $\hat{Y}$ to be upper-bounded by d. For reasons of scalability, this matrix should never be explicitly maintained. Its entries can be generated dynamically from the user and item factors as needed.

In MF, the factors are learned by minimizing the squared Euclidean distance between the actual ratings and the predicted ratings as:

$$\min_{U,V} \sum_{i \in \mathcal{U}, j \in \mathcal{V}} \frac{1}{2} w_{ij} (Y_{ij} - \hat{Y}_{ij})^2. \tag{2}$$

For recommender systems, making the model predictions close to the ratings is only a means to an end. The true objective is to facilitate the ordering of the items according to the users preference. On one hand the objective (2) is solving a harder problem, and on the other hand, in its goal to keep the predicted $\hat{Y}_{ij}$ *point-wise* close to the observed Y_{ij} the regressor may generate predictions close in squared Euclidean sense with estimated scores that induce an incorrect order, thereby misdirecting the algorithm [9]. This is particularly true when one uses a regressor of limited function-fitting capacity. In this context the cost function (2) is unnecessarily strict.

These drawbacks are now well recognized and have inspired several ranking based approaches to the the recommendation problem. The newer approaches replace (2) by other cost functions that depend not on the predicted scores themselves, but on the order induced by them. Examples of such cost functions include the normalized cumulative discounted gain (NDCG), expected reciprocal return (ERR) and mean absolute precision (MAP) [18]. However, the domain of these cost functions is the space of permutations of the list of items. This is not only a discrete (combinatorial) space, but also one whose size grows exponentially with the number of items. This makes training with respect to these cost functions difficult.

In this paper we adapt a list-wise learning to rank (LETOR) algorithm called monotone retargeting (MR) [2], developed recently by the authors, to the recommendation task. MR introduces a new family of cost functions that have desirable computational and statistical properties. It uses a cost function that is truly a function of the order and not of the predicted values, therefore well suited for ranking. The key observations that motivates the MR is that (i) the combinatorial problem of LETOR can be transformed into a problem of searching over the space of all monotonic transformations and (ii) it is possible to design an efficient, convergent technique to solve the challenging task of minimizing over this infinite function space without sacrificing generality. A pictorial representation of MR is shown in Fig. 1.

2.1 Retargeted Matrix Factorization

Given any loss function $D : \mathbb{R} \times \mathbb{R} \mapsto \mathbb{R}_+$, we may define a pointwise LETOR based MF problem by:

$$\min_{U,V} \sum_i \sum_j w_{ij} D(Y_{ij}, f(\langle \boldsymbol{u}_i, \boldsymbol{v}_j \rangle))$$

where $f : \mathbb{R} \mapsto \mathbb{R}$ is some regression function with parameters $\boldsymbol{u}_i$ and $\boldsymbol{v}_j$. As discussed earlier, this is unnecessarily stringent for ranking. A conceptually better alternative is:

$$\min_{U,V,\{\Upsilon_i \in \mathcal{M}\}} \sum_i \sum_j w_{ij} D\big(Y_{ij}, \Upsilon_i \circ f(\langle \boldsymbol{u}_i, \boldsymbol{v}_j \rangle)\big),$$

where $\Upsilon_i : \mathbb{R} \mapsto \mathbb{R}$ is a monotonic increasing transformation and $\mathcal{M}$ is the class of all such transformations. With no loss in generality of modeling, we may apply the monotonic transform to Y_{ij} instead. This avoids the minimization over the function composition, but the need for minimizing over the set of all monotone functions remains.

As noted in [2] the optimization over the infinite space of functions $\mathcal{M}$ can be converted into one over finite dimensional vector spaces provided we have a finite characterization of the constraint set $\mathcal{R}\!\downarrow_i$. Without loss of generality, the monotone transformation can be applied to the left hand side i.e. to the sorted ratings. Let $Y_{i,*}$ represent the item ratings for user i arranged in (non-unique) sorted order. Further, let $V_i \in \mathbb{R}^{|\mathcal{V}_i| \times d}$ represent the subset of item factors corresponding to the items rated by user i, sorted to match the ratings $Y_{i,*}$. The resulting cost function is:

$$\min_{U,V,\{\boldsymbol{r}_i \in \mathcal{R}\downarrow_i\}} \sum_i D(\boldsymbol{r}_i, f(V_i \boldsymbol{u}_i)) \tag{3}$$
$$\text{where } \mathcal{R}\!\downarrow_i = \{\boldsymbol{r}|_{M(Y_{i,*})=\boldsymbol{r}}^{\exists M \in \mathcal{M}}\} \; \forall i,$$

$f(\cdot)$ is applied element-wise to result of the matrix vector product $V_i \boldsymbol{u}_i$, and $\mathcal{R}\!\downarrow_i$ represents all vectors that are sorted in decreasing order. Hence $\mathcal{R}\!\downarrow_i$ includes vectors $\boldsymbol{r} \in \mathbb{R}^{|\mathcal{V}_i|}$ such that $r_{k+1} = r_k$ for some k. Since the vectors $\boldsymbol{r}$ are the targets given to the regressor to fit, it is more robust to enforce separation between the components, i.e. maintain $r_k \geq r_{k+1} + \epsilon$. We indicate the set of all such ϵ separated decreasing ordered sets by $\mathcal{R}_\epsilon\!\downarrow_i$. We now characterize these sets.

The Set $\mathcal{R}\!\downarrow_i$: The convex composition $\boldsymbol{r} = \alpha \boldsymbol{r}_1 + (1-\alpha)\boldsymbol{r}_2$ of two isotonic vectors $\boldsymbol{r}_1$ and $\boldsymbol{r}_2$ preserves isotonicity, as does the scaling $\alpha \boldsymbol{r}$ for any $\alpha \in \mathbb{R}_+$. Hence the set $\mathcal{R}\!\downarrow_i$ is a convex cone. Clearly the same holds for $\mathcal{R}_\epsilon\!\downarrow_i$. Convex-conicity makes the problem computationally tractable because the set can be described entirely by its extreme rays, or by the extreme rays of its polar. $\mathcal{R}\!\downarrow_i$ can be expressed as the image of the set $\{\mathbb{R}_+\}^{|\mathcal{V}_i|-1} \times \mathbb{R}$ under a linear transformation by a particular upper triangular matrix S with positive entries. The matrix S is not unique and can be generated from any vector $\boldsymbol{v} \in \mathbb{R}_+{}^{|\mathcal{V}_i|}$, but since any member from the allowed class of S is sufficient for a *exhaustive* representation of $\mathcal{R}\!\downarrow_i$ we use the vector $\mathbf{1}$ to generate S. The property is stated formally in the following lemma:

LEMMA 1. *The set $\mathcal{R}\!\downarrow_i$ of all vectors in $\mathbb{R}^{|\mathcal{V}_i|}$ that are sorted in a descending order is given by $S\boldsymbol{x}$ s.t. $\boldsymbol{x} \in \{\mathbb{R}_+\}^{|\mathcal{V}_i|-1} \times \mathbb{R}$ where S is a triangular matrix generated from a vector $\mathbf{1}$ such that the k^{th} row $S(k,:)$ is $\{0\}^{k-1} \times \mathbf{1}(k:)$*

The proof of Lemma 1 appears in [2]. In addition, the following lemma defining $\mathcal{R}_\epsilon\!\downarrow_i$ follows directly from Lemma 1 and is stated without proof.

LEMMA 2. *The set $\mathcal{R}_\epsilon\!\downarrow_i$ of all vectors in $\mathbb{R}^{|\mathcal{V}_i|}$ that are sorted in a descending order and whose consecutive components are separated by ϵ is given by $S\boldsymbol{x}$ s.t. $\boldsymbol{x} \in \{\mathbb{R}_\epsilon\}^{|\mathcal{V}_i|-1} \times \mathbb{R}$ where S is a triangular matrix generated from a vector $\mathbf{1}$ such that the k^{th} row $S(k,:)$ is $\{0\}^{k-1} \times \mathbf{1}(k:)$*

In addition to these sets we shall make frequent use of the set of all discrete probability distributions that are in descending order, i.e. $\mathcal{R}\!\downarrow_i \cap \Delta_i$ that we represent by $\Delta\!\downarrow_i$. We give a similar representation of this set by generating an upper triangular matrix T from the vector $\boldsymbol{v}_{\Delta_i} = \{1, \frac{1}{2}, \cdots \frac{1}{i} \cdots \frac{1}{|\mathcal{V}_i|}\}$ and considering $\boldsymbol{x} \in \Delta_i$. This is formally stated in the following lemma:

LEMMA 3. *The set $\Delta\!\downarrow_i$ of all discrete probability distributions of dimension $|\mathcal{V}_i|$ that are in descending order is the image $T\boldsymbol{x}$ such that $\boldsymbol{x} \in \Delta_i$ where T is an upper triangular matrix generated from the vector $\boldsymbol{v}_{\Delta_i} = \{1, \frac{1}{2} \cdots \frac{1}{|\mathcal{V}_i|}\}$ such that $T(k,:) = \{0\}^{k-1} \times \boldsymbol{v}_\Delta(k:)$*

The proof of Lemma 3 appears in [2]. Similar to the set $\mathcal{R}_\epsilon\!\downarrow_i$ we will use a well separated version of $\Delta\!\downarrow_i$ denoted by $\Delta_\epsilon\!\downarrow_i$ that consists of discrete probability distributions sorted in decreasing order but also satisfying $r_i > r_{i+1} + \frac{\epsilon}{i}$. Note that vectors in $\Delta_\epsilon\!\downarrow_i$ are not only sorted but have an increasing gap between consecutive components. The motivation for such a construction is that even if the regressor is inaccurate, if the inaccuracy is more or less uniform across the components, the top entries of the predictions will be less prone to order reversal than the bottom ones: a desirable feature for ranking. The following lemma formally states the representation for $\Delta_\epsilon\!\downarrow_i$.

LEMMA 4. *The set $\Delta_\epsilon\!\downarrow_i$ of all discrete probability distributions of dimension $|\mathcal{V}_i|$ that are in descending order is the image $T\boldsymbol{x}$ s.t. $\boldsymbol{x} \in \Delta_{\epsilon_i}$ where T is an upper triangular matrix generated from the vector $\boldsymbol{v}_{\Delta_i} = \{1, \frac{1}{2} \cdots \frac{1}{|\mathcal{V}_i|}\}$ such that $T(k,:) = \{0\}^{k-1} \times \boldsymbol{v}_\Delta(k:)$*

The prof of Lemma 4 follows directly from Lemma 3.

2.2 Cost function

With appropriate choices of the distance like function $D(\cdot,\cdot)$ and the curve fitting function $f(\cdot)$ we can transform (3) into a bi-convex optimization problem over a product of convex sets. We choose $D(\cdot,\cdot)$ to be a Bregman divergence $D_\phi\big(\cdot\big|\big|\cdot\big)$ as defined in Section 1.2, and $f(V_i\boldsymbol{u}_i) = (\nabla\phi)^{-1}(V_i\boldsymbol{u}_i)$. To simplify the notation, we define $C_\phi\big(\boldsymbol{a}\big|\big|\boldsymbol{b}\big) = D_{\boldsymbol{\phi}}\big(\boldsymbol{a}\big|\big|(\nabla\phi)^{-1}(\boldsymbol{b})\big)$. The resulting cost function is given by:

$$\min_{U,V,\{\boldsymbol{r}_i \in \mathcal{R}\downarrow_i\}} \sum_i C_\phi\big(\boldsymbol{r}_i\big|\big|V_i\boldsymbol{u}_i\big). \tag{4}$$

Let $\boldsymbol{y}_i = (\nabla\phi)^{-1}(V_i\boldsymbol{u}_i)$ with represent the vector of predicted ratings of user i. Using Legendre duality one recognizes that equation (4) quantifies the gap in the Fenchel-Young inequality[2]

$$D_{\boldsymbol{\phi}}\big(\boldsymbol{r}_i\big|\big|(\nabla\phi)^{-1}(\boldsymbol{y}_i)\big) = \psi(\boldsymbol{y}_i) + \phi(\boldsymbol{r}_i) - \langle \boldsymbol{r}_i, \boldsymbol{y}_i\rangle\,,$$

where ψ is the Legendre conjugate of ϕ. Although this clarifies the issue of separate convexity in $\boldsymbol{r}_i$ and $\boldsymbol{y}_i$, the conditions under which joint convexity is obtained are not obvious.

2.3 Joint Convexity and Global Minimum

Joint convexity, if ensured, guarantees global minimum even for a coordinate-wise minimization because our constraint set is a product of convex sets. This important question is resolved in the following theorem from [2] (Theorem 2).

THEOREM 1. *The gap in the Fenchel-Young inequality: $\psi(\boldsymbol{y}) + \phi(\boldsymbol{x}) - \langle \boldsymbol{x}, \boldsymbol{w}\rangle$ for any twice differentiable strictly convex $\phi(\cdot)$ with a differentiable conjugate $(\phi)^*(\cdot) = \psi(\cdot)$ is jointly convex if and only if $\phi(\boldsymbol{x}) = c||\boldsymbol{x}||^2$ for all $c > 0$.*

We note that since we maintain explicit representation of the factor matrices U and V, the optimization problem is no longer convex with respect to these factors. However, the following proposition from [1] (Proposition 5) shows conditions under which all local minima in terms of U and V are global, and correspond to the same regression matrix $\hat{Y} = UV^\top$.

PROPOSITION 1. *Let G be a twice differentiable convex function on matrices of size $p \times q$ with compact level sets. Let $d > 1$ and $(U,V) \in \mathbb{R}^{p\times d} \times \mathbb{R}^{q\times d}$ a local optimum of the function $H : \mathbb{R}^{p\times d} \times \mathbb{R}^{q\times d} \mapsto \mathbb{R}$ defined by $H(U,V) = G(UV^\top)$, that is U such that $\nabla H(U,V) = 0$ and the Hessian of H at (U,V) is positive semi-definite. If U or V is rank deficient, then $N = UV^\top$ is a global optimum of G, that is $\nabla G(N) = 0$.*

Combining Theorem 1 and Proposition 1, it follows that under mild conditions, R-MF using squared loss recovers a unique solution. Proposition 1 applied to other Bregman divergences can only provide local optimality guarantees.

2.4 Optimality of Sorting

For any sorted vector $\boldsymbol{r}$, finding the permutation of $\boldsymbol{y}$ that minimizes $D_\phi\big(\boldsymbol{r}\big|\big|\boldsymbol{y}\big)$ shows up as a sub-problem in our formulation that needs to be solved in an inner loop. Thus solving it efficiently is critical and this is yet another instance where Bregman divergences are very useful.

For an arbitrary divergence function the search over the optimal permutation is a *non-linear assignment* problem that can be solved only by exhaustive enumeration. For an arbitrary separable divergence the optimal permutation may be found by solving a linear assignment problem, which is an integer linear program and hence also expensive to solve (especially in an inner loop, as required in our algorithm).[3]

On the other hand, if $\boldsymbol{\phi}(\cdot)$ is IS, the solution is remarkably simple, as shown in Lemma 5 where $\left[\begin{smallmatrix} r_1 \\ r_2 \end{smallmatrix}\right]$ denotes a vector in $\mathbb{R}^2$ with components r_1 and r_2.

[2]The **Fenchel-Young** inequality:

$$\psi(\boldsymbol{w}) + \phi(\boldsymbol{x}) - \langle \boldsymbol{w}, \boldsymbol{x}\rangle \geq 0.$$

[3]One of our baseline tools, COFI^RANK-NDCG [25] does need to solve such an assignment problem in each iteration. Speed comparisons are included in Section 4.

LEMMA 5. *If $r_1 \geq r_2$ and $y_1 \geq y_2$ and $\phi(\cdot)$ is IS, then:*

$$D_\phi\left(\left[\begin{smallmatrix}r_1\\r_2\end{smallmatrix}\right]\middle\|\left[\begin{smallmatrix}y_1\\y_2\end{smallmatrix}\right]\right) \leq D_\phi\left(\left[\begin{smallmatrix}r_1\\r_2\end{smallmatrix}\right]\middle\|\left[\begin{smallmatrix}y_2\\y_1\end{smallmatrix}\right]\right) \quad \text{and}$$
$$D_\phi\left(\left[\begin{smallmatrix}y_1\\y_2\end{smallmatrix}\right]\middle\|\left[\begin{smallmatrix}r_1\\r_2\end{smallmatrix}\right]\right) \leq D_\phi\left(\left[\begin{smallmatrix}y_2\\y_1\end{smallmatrix}\right]\middle\|\left[\begin{smallmatrix}r_1\\r_2\end{smallmatrix}\right]\right).$$

Lemma 5 is trivially extended by induction to the vector case.

3. FORMULATION AND ALGORITHM

We safeguard against overfitting by adding squared Frobenius norm regularization for the matrices U and V. Note that the cost function (4) is not invariant to scale. For example squared Euclidean distance and KL divergence are homogeneous functions of degree 2 and 1 respectively. Thus the cost can be reduced just by scaling its arguments down, without actually learning the task. To remedy this, we restrict the $\boldsymbol{r}_i$'s from shrinking below a predefined size. This is accomplished by constraining $\boldsymbol{r}_i$'s to lie in an appropriate closed convex set not only separated from the origin but also to a set of vectors whose adjacent components are separated from each other. For the latter we use the set $\Delta_{\epsilon\downarrow}$ as defined before. After these modifications we obtain the final formulation[4] as:

$$\min_{U,V,\{\boldsymbol{r}_i \in \Delta_{\epsilon\downarrow i}\}} \sum_i C_\phi\left(\boldsymbol{r}_i \middle\| V_i \boldsymbol{u}_i\right) + \frac{\lambda_u}{2}\sum_i ||\boldsymbol{u}_i||^2 + \frac{\lambda_v}{2}\sum_j ||\boldsymbol{v}_j||^2 \quad (5)$$

In our formulation we assume that the true movie ratings are totally ordered, though the finer ordering between similar items is not visible to the ranking algorithm. Let $P_j = \{P_{jk}\}_{k=1}^{k_j}$ be a partition of the index set of $\mathcal{V}_j$, such that all items in P_{jk} have the same training score. (for example the set of all movies rated 5 by a particular user). The sets $\mathcal{V}_j$ effectively get partitioned further into $\{P_{jk}\}_{k=1}^{k_j}$. Though the ratings provide an order between movies from any two different sets P_{jk} and P_{jl}, the order within any set P_{jk} remains unknown. To retrieve the total order we introduce a block-diagonally restricted permutation matrix $\mathbb{P}_j$ that can permute indices in each P_{jk} independently. Since the items in P_{jk} are not equivalent they are available for re-ordering as long as that minimizes the cost (5). IS Bregman divergences have the special property that sorting minimizes the divergence over all permutations (Lemma 5). Thus update (6) can be accomplished by sorting.

The combined algorithm for R-MF is given in Fig. 2. We cycle through all three update steps until convergence. The update (7) can be solved using any of a number of constrained convex optimization algorithms. We implemented (7) using the exponentiated gradient (EG) algorithm [12], a proximal gradient method for simplex constrained vectors. EG requires simple multiplicative updates using the function gradient followed by a normalization step. Update (6) is a parallel sort. The user and item factors (8) can be updated using any number of matrix factorization algorithms [22].

4. EXPERIMENTS

[4]To apply the results of Proposition 1 to the regularized case, we will require the variational representation of the matrix trace norm $\|\cdot\|_*$ under similar rank deficient conditions for the mapping between the cost function $G(N)$ and $H(U,V)$:

$$\|N\|_* = \min_{U,V \mid N=UV^\top} \frac{1}{2}\left(\sum_i ||\boldsymbol{u}_i||^2 + \sum_j ||\boldsymbol{v}_j||^2\right)$$

$$\mathbb{P}_i^{t+1} = \operatorname*{Argmin}_{\pi} C_\phi\left(T\boldsymbol{x}_i^t \middle\| \pi V_i \boldsymbol{u}_i\right) \quad \forall i \text{ in parallel} \quad (6)$$
$$\boldsymbol{x}_i^{t+1} = \operatorname*{Argmin}_{\boldsymbol{x} \in \Delta_{\epsilon\downarrow}} C_\phi\left(T\boldsymbol{x} \middle\| \mathbb{P}_i^{t+1} V_i \boldsymbol{u}_i\right) \quad \forall i \text{ in parallel} \quad (7)$$
$$U^{t+1}, V^{t+1} = \operatorname*{Argmin}_{U,V} \sum_i C_\phi\left(T\boldsymbol{x}_i^{t+1} \middle\| \mathbb{P}_i^{t+1} V \boldsymbol{u}\right) + \frac{\lambda_u}{2}\sum_i ||\boldsymbol{u}_i||^2 + \frac{\lambda_v}{2}\sum_j ||\boldsymbol{v}_j||^2 \quad (8)$$

Figure 2: Algorithm for optimizing R-MF

Table 2: Data sizes after preprocessing to remove users with less than 30 ratings

Dataset	# Users	# Items	# Ratings
Movielens 100K	744	1,682	95,269
Movielens 1M	5,289	3,701	982,040
Movielens 10M	57,534	10,675	9,704,223
Flixster	30,277	48,146	7,580,563

We evaluated R-MF on four publicly available recommendation datasets:

- Movielens[5] is a movie recommendation website administered by GroupLens Research. GroupLens Research has made available three ratings datasets of varying sizes[6]. We used the movielens 100K, movielens 1M and movielens 10M datasets. Ratings in Movielens 100K and movielens 1M take one of 5 values in the set $\{1.0, 2.0, \ldots, 5.0\}$. Ratings in movielens 10M take one of 10 values in the set $\{0.5, 1, 1.5, \ldots, 5.0\}$
- Flixster[7] is a website where users share film reviews and ratings. We used the flixster dataset provided by [11] with timestamps. Ratings in Flixster take one of 10 values in the set $\{0.5, 1, 1.5, \ldots, 5.0\}$.

Data preprocessing: First, we removed all users with less than 30 ratings. Each of the evaluated datasets contains the time-stamp of the user rating. For each user, we selected the last third of the ratings sorted by user-time as the test set, the middle third as the validation set, and any left over ratings were selected as the training set. This partitioning scheme ensures that each user has at least 10 ratings in the validation and test sets so we are able to compute the top-k performance metrics for at least 10 retrieved items per user. Details of the dataset sizes after preprocessing are provided in table Table 2.

We experimented with R-MF using squared loss motivated by the optimization theoretic guarantees discussed in Section 2.3. Further experiments with other Bregman divergences will be implemented in an extended version of this manuscript. We also evaluated the performance of COFI$^{\text{RANK}}$-NDCG [25] and COFI$^{\text{RANK}}$-ordinal [26] as baseline models using the C++ implementation provided by the authors[8]. For all models, we selected the regularization parameter $\lambda = \lambda_u = \lambda_v$ from the set $10^{\{-2,-1.5,-1,\ldots,2\}}$.

The models are scored using metrics commonly used for evaluating ranking models [18]. We plot normalized discounted cumulative gain (NDCG) and precision in Fig. 3 while varying the number of retrieved items $k = \{1, \ldots 10\}$. Further results with

[5]movielens.umn.edu
[6]www.grouplens.org/node/73
[7]www.flixster.com
[8]available at www.cofirank.org

Table 3: Expected reciprocal return (ERR) results on recommender system datasets. "–" represents datasets where CoFiRank -NDCG did not finish after running for more than seven days.

	R-MF	CoFiRank -NDCG	CoFiRank -ORD.
Movielens 100K			
Rank 10	**0.779**	0.702	0.722
Rank 20	**0.777**	0.697	0.709
Movielens 1M			
Rank 10	**0.819**	0.726	0.774
Rank 20	**0.816**	0.728	0.756
Movielens 10M			
Rank 10	**0.780**	0.687	0.763
Rank 20	**0.781**	–	0.753
Flixster			
Rank 10	**0.771**	–	0.743
Rank 20	**0.771**	–	0.737

other ranking metrics were also computed including expected reciprocal return (ERR) in Table 3, mean average precision (MAP) in Table 4, and NDCG of the full list in Table 5. For the precision and MAP metrics, a movie was labelled as relevant if its rating was greater than 4.

Our experiments show that R-MF improves ranking performance over CoFiRank -NDCG and CoFiRank -ordinal as measured by all the metrics that we computed. We note that CoFiRank -ordinal [26] has been shown to outperform several state of the art models including maximum margin matrix factorization [23] and Gaussian process ordinal regression [7]. The results were even more striking when we compared the NDCG performance of R-MF to CoFiRank -NDCG, though the algorithm is specifically designed to optimize NDCG. Our results confirm the observation of other authors, including the authors of CoFiRank that CoFiRank -ordinal seems to out-perform CoFiRank -NDCG. It is unclear why this is the case. We found that CoFiRank to be susceptible to overfitting when we compared the test performance with the training NDCG values. This might explain some of the performance gap. Our results were qualitatively very similar for rank 10 and rank 20 models.

R-MF was implemented in Python/Numpy. Cython was used to implement the parallel retargeting updates (7) and parallel sorting (6). The matrix factorization step (8) was solved using alternating least squares. The code was executed on a 2.4GHz quad-core Intel Xeon processor. Timing on the larger movie datasets are shown in Table 6 and compared to CoFiRank . We found that R-MF exhibited much better scaling behavior as the size data increased. We suspect that the large observed runtimes of CoFiRank -NDCG are due to the cost of the linear assignment problem that must be solved for each user at every iteration. The linear assignment can be solved using a number of efficient algorithms [16], but the computational requirements scale cubically with the number of ratings per user. R-MF is able to avoid solving this linear assignment problem as we prove that sorting recovers the optimal ordering (Section 2.4).

Our experience suggests that the speed of the method can be significantly improved by (i) implementation using a faster language e.g c++ (ii) matrix factorization using stochastic gradient descent or other scalable solvers (iii) interleaving the factorization and retargeting steps, as the matrix factorization is the most computationally intensive portion of the algorithm. We leave such implementation improvements to future work.

Table 4: Mean absolute precision (MAP) results on recommender system datasets. Relevant ratings have a value greater than 4. "–" represents datasets where CoFiRank -NDCG did not finish after running for more than seven days.

	R-MF	CoFiRank -NDCG	CoFiRank -ORD.
Movielens 100K			
Rank 10	**0.425**	0.335	0.364
Rank 20	**0.427**	0.333	0.348
Movielens 1M			
Rank 10	**0.485**	0.358	0.435
Rank 20	**0.482**	0.360	0.408
Movielens 10M			
Rank 10	**0.472**	0.353	0.453
Rank 20	**0.470**	–	0.439
Flixster			
Rank 10	**0.513**	–	0.488
Rank 20	**0.509**	–	0.480

Table 5: NDCG Results on recommender system datasets. "–" represents datasets where CoFiRank -NDCG did not finish after running for more than seven days.

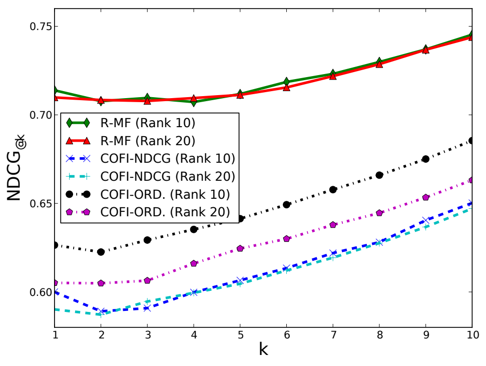
	R-MF	CoFiRank -NDCG	CoFiRank -ORD.
Movielens 100K			
Rank 10	**0.883**	0.839	0.854
Rank 20	**0.882**	0.838	0.844
Movielens 1M			
Rank 10	**0.896**	0.846	0.877
Rank 20	**0.895**	0.847	0.867
Movielens 10M			
Rank 10	**0.892**	0.843	0.883
Rank 20	**0.891**	–	0.878
Flixster			
Rank 10	**0.888**	–	0.877
Rank 20	**0.887**	–	0.874

Table 6: Average training time (mins) on the larger recommender system datasets. "–" represents datasets where CoFiRank -NDCG did not finish after running for more than seven days. The Movielens 10M (rank 20) result of CoFiRank -NDCG is based on the average runtime for a subset of the parameters, as the full parameter sweep did not finish running.

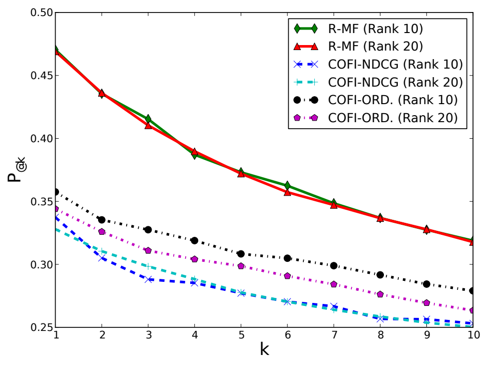
	R-MF	CoFiRank -NDCG	CoFiRank -ORD.
Movielens 1M			
Rank 10	121.025	**329.889**	41.641
Rank 20	259.051	**340.569**	30.276
Movielens 10M			
Rank 10	396.900	**2912.179**	1391.249
Rank 20	663.889	**7609.683**	820.579
Flixster			
Rank 10	1657.314	**–**	2459.418
Rank 20	2203.207	**–**	897.989

5. CONCLUSION AND FUTURE WORK

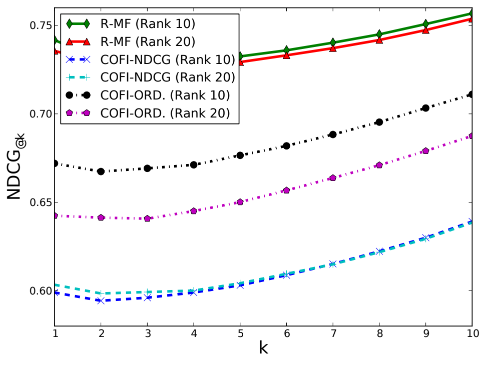
In this paper, we proposed retargeted matrix factorization (R-MF), a novel approach for learning the user-wise ranking of items for collaborative filtering. Retargeted matrix factorization improves



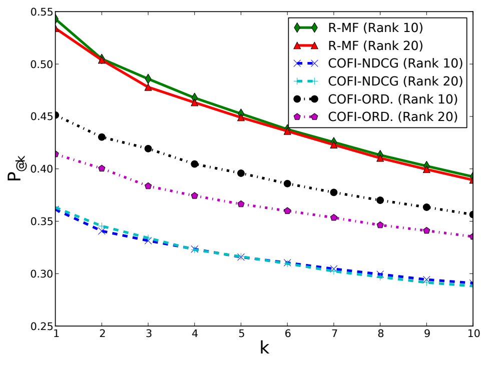
(a) Movielens 100K NDCG



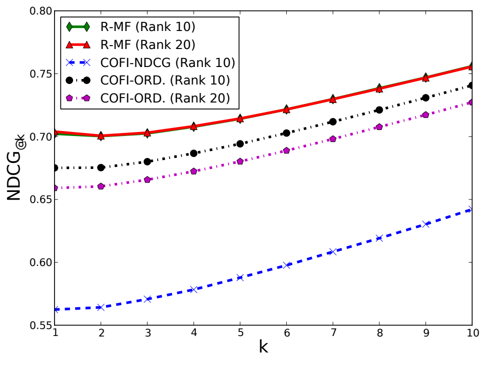
(b) Movielens 100K Precision



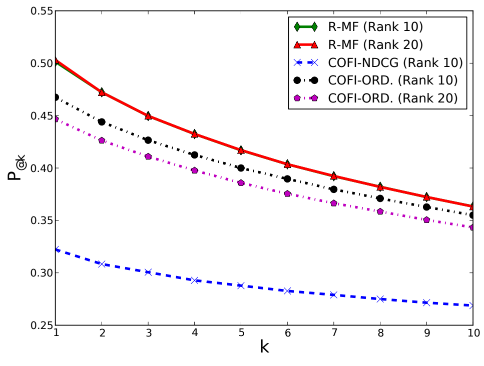
(c) Movielens 1M NDCG



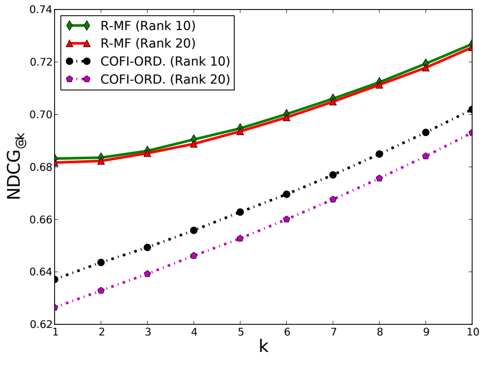
(d) Movielens 1M Precision



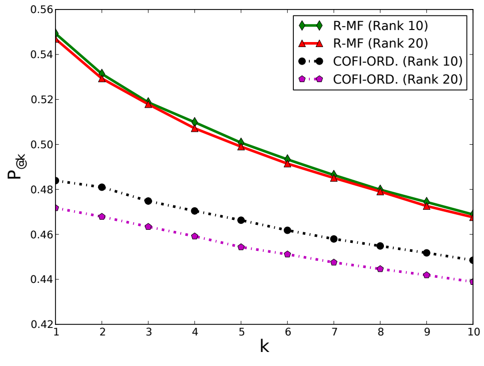
(e) Movielens 10M NDCG



(f) Movielens 10M Precision



(g) Flixster NDCG



(h) Flixster Precision

Figure 3: Performance results: $\mathrm{NDCG}_{@k}$ and $\mathrm{P}_{@k}$ at $@k = 1, 2, \ldots, 10$. Relevant ratings have a value greater than 4. COFI$^{\text{RANK}}$ - ordinal [26] has been shown to outperform several state of the art models including maximum margin matrix factorization [23] and Gaussian process ordinal regression [7].

the ranking performance by searching for a monotonic transformation of the ratings that results in a better fit while preserving their ranked order. R-MF was compared to the ranking and ordinal regression variants of CofiRank and evaluated on benchmark movie recommendation datasets. Our results show that retargeted matrix factorization results in superior ranking performance compared to CofiRank, though CofiRank is specifically designed to optimize NDCG. Our results also highlight the scalability of R-MF in comparison to CofiRank. We plan to explore scalability and implementation concerns in more detail in future work. Although the unique solution is only guaranteed for squared loss, the performance results presented in [2] provide some motivation for experimenting with other Bregman divergences in future work.

6. ACKNOWLEDGMENTS

Authors acknowledge support from NSF grant IIS 1016614

7. REFERENCES

[1] J. Abernethy, F. Bach, T. Evgeniou, and J.-P. Vert. A new approach to collaborative filtering: Operator estimation with spectral regularization. *JMLR*, 10:803–826, 2009.

[2] S. Acharyya, O. Koyejo, and J. Ghosh. Learning to rank with Bregman divergences and monotone retargeting. In *Proceedings of the 28th conference on Uncertainty in artificial intelligence*, UAI '12, 2012.

[3] S. Balakrishnan and S. Chopra. Collaborative ranking. In *Proceedings of the fifth ACM international conference on Web search and data mining*, WSDM '12, pages 143–152, New York, NY, USA, 2012. ACM.

[4] A. Banerjee, S. Merugu, I. Dhillon, and J. Ghosh. Clustering with Bregman divergences. *Journal of Machine Learning Research*, 6:1705–1749, 2005.

[5] J. Basilico and T. Hofmann. Unifying collaborative and content-based filtering. In *ICML '04: Proceedings of the twenty-first international conference on Machine learning*, page 9, New York, NY, USA, 2004. ACM.

[6] Z. Cao, T. Qin, T.-Y. Liu, M.-F. Tsai, and H. Li. Learning to rank: from pairwise approach to listwise approach. In *ICML '07: Proceedings of the 24th international conference on Machine learning*, pages 129–136, New York, NY, USA, 2007. ACM.

[7] W. Chu and Z. Ghahramani. Gaussian processes for ordinal regression. *Journal of Machine Learning Research*, 6:2005, 2004.

[8] K. Crammer and Y. Singer. Pranking with ranking. In *Advances in Neural Information Processing Systems 14*, pages 641–647. MIT Press, 2001.

[9] P. Cremonesi, Y. Koren, and R. Turrin. Performance of recommender algorithms on top-n recommendation tasks. In *Proceedings of the fourth ACM conference on Recommender systems*, RecSys '10, pages 39–46. ACM, 2010.

[10] Y. Freund, R. Iyer, R. E. Schapire, and Y. Singer. An efficient boosting algorithm for combining preferences. *J. Mach. Learn. Res.*, 4:933–969, 2003.

[11] M. Jamali and M. Ester. A matrix factorization technique with trust propagation for recommendation in social networks. In *RecSys*, pages 135–142, 2010.

[12] J. Kivinen and M. K. Warmuth. Exponentiated gradient versus gradient descent for linear predictors. *Information and Computation*, 132, 1995.

[13] Y. Koren, R. Bell, and C. Volinsky. Matrix factorization techniques for recommender systems. *Computer*, 42:30–37, 2009.

[14] Y. Koren and J. Sill. Ordrec: an ordinal model for predicting personalized item rating distributions. In B. Mobasher, R. D. Burke, D. Jannach, and G. Adomavicius, editors, *RecSys*, pages 117–124. ACM, 2011.

[15] O. Koyejo and J. Ghosh. A kernel-based approach to exploiting interaction-networks in heterogeneous information sources for improved recommender systems. In *HetRec '11*, pages 9–16, 2011.

[16] H. Kuhn. The Hungarian method for the assignment problem. *Naval research logistics quarterly*, 2(1-2):83–97, 1955.

[17] N. N. Liu, M. Zhao, and Q. Yang. Probabilistic latent preference analysis for collaborative filtering. In *Proceedings of the 18th ACM conference on Information and knowledge management*, CIKM '09, pages 759–766, New York, NY, USA, 2009. ACM.

[18] T. Liu. *Learning to Rank for Information Retrieval*. Information retrieval. Springer, 2011.

[19] P. Ravikumar, A. Tewari, and E. Yang. On NDCG consistency of listwise ranking methods. In *Proceedings of 14th International Conference on Artificial Intelligence and Statistics*, AISTATS, 2011.

[20] R. T. Rockafellar. *Convex Analysis (Princeton Landmarks in Mathematics and Physics)*. Princeton University Press, December 1996.

[21] Y. Shi, M. Larson, and A. Hanjalic. List-wise learning to rank with matrix factorization for collaborative filtering. In *Proceedings of the fourth ACM conference on Recommender systems*, RecSys '10, pages 269–272, New York, NY, USA, 2010. ACM.

[22] A. P. Singh and G. J. Gordon. A unified view of matrix factorization models. In *Machine Learning and Knowledge Discovery in Databases, European Conference (ECML/PKDD)*, 2008.

[23] N. Srebro, J. D. M. Rennie, and T. S. Jaakola. Maximum-margin matrix factorization. In *Advances in Neural Information Processing Systems 17*, pages 1329–1336. MIT Press, 2005.

[24] H. Steck. Training and testing of recommender systems on data missing not at random. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '10, pages 713–722, New York, NY, USA, 2010. ACM.

[25] M. Weimer, A. Karatzoglou, Q. V. Le, and A. Smola. Cofirank, maximum margin matrix factorization for collaborative ranking. In *NIPS*, volume 20, 2007.

[26] M. Weimer, A. Karatzoglou, and A. J. Smola. Improving maximum margin matrix factorization. *Machine Learning*, 72(3):263–276, 2008.